Boxify : Sales Analysis and Inventory Insights – Project - By Saloni Prasad

GIT HUB LINK - https://github.com/Saloniprasad17/Boxify-Sales-Analysis-and-Inventory-Insights-Project.git

Instruction

Effective inventory management is essential for businesses to maintain optimal stock levels, minimize carrying costs, and meet customer demand. As a data analyst, your task is to analyze a sales dataset, extract valuable insights, and provide inventory-driven recommendations to enhance inventory management practices.

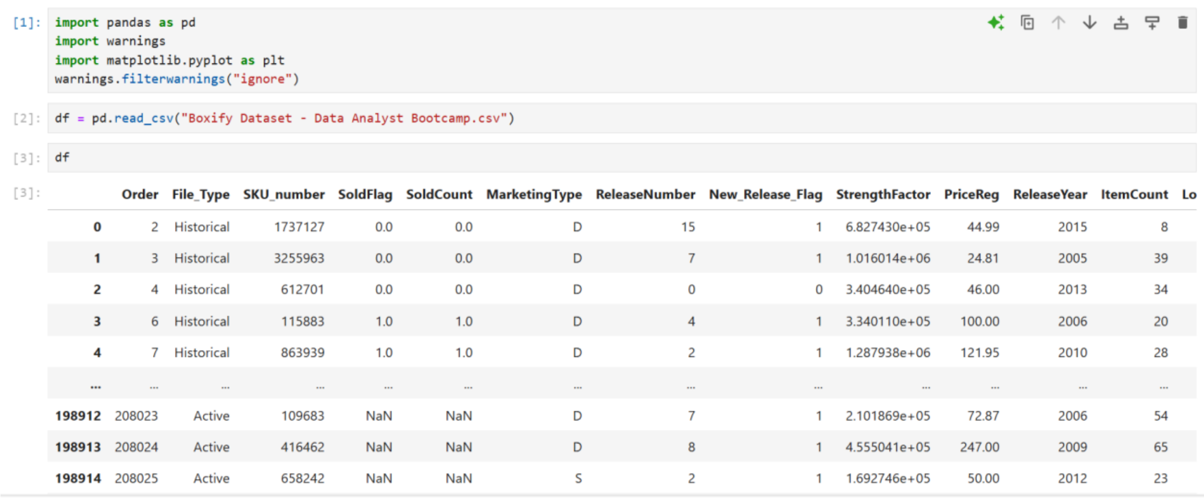
Objectives:

1. Analyze the provided sales dataset to understand sales trends, stock levels, and product performance.
2. Identify popular products, low-stock items, and sales patterns over time.
3. Generate actionable recommendations for improving inventory management efficiency.

Tasks/Activities List:

1. Data Collection and Preprocessing:
    - Obtain the sales dataset from the provided source: Sales Analysis Dataset.

- Clean and preprocess the data to handle missing values and inconsistencies.



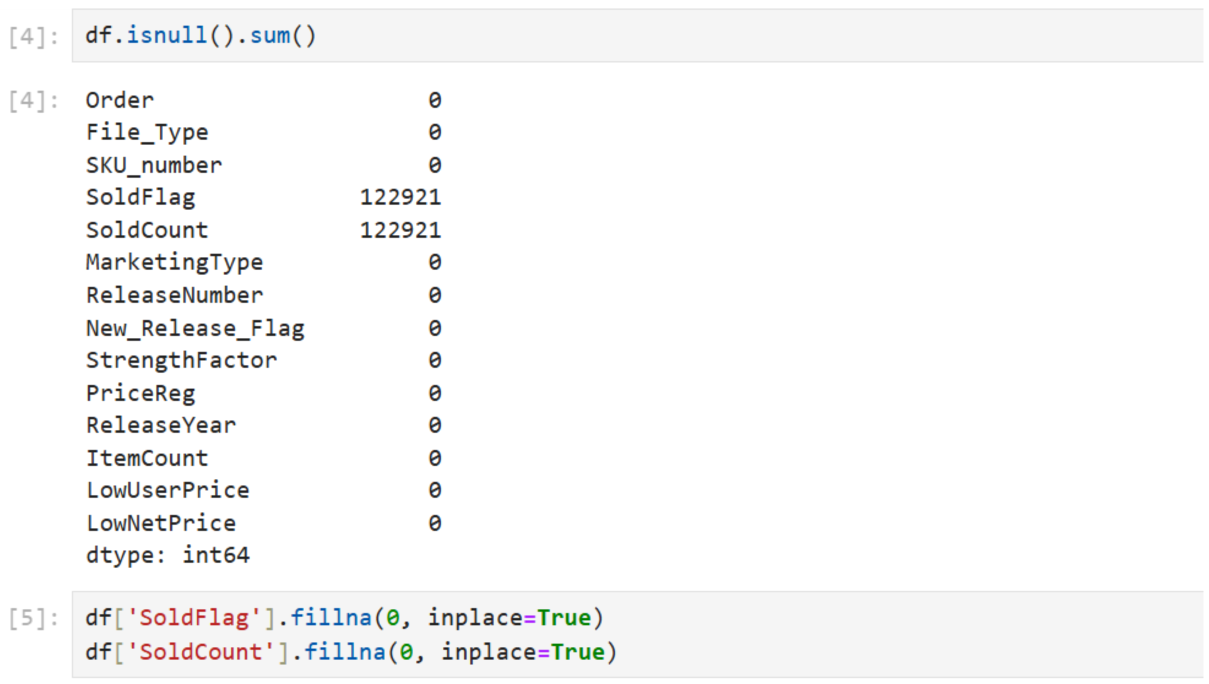
```
[1]: import pandas as pd
     import warnings
     import matplotlib.pyplot as plt
     warnings.filterwarnings("ignore")
```

```
[2]: df = pd.read_csv("Boxify Dataset - Data Analyst Bootcamp.csv")
```

```
[3]: df
```

[3]:

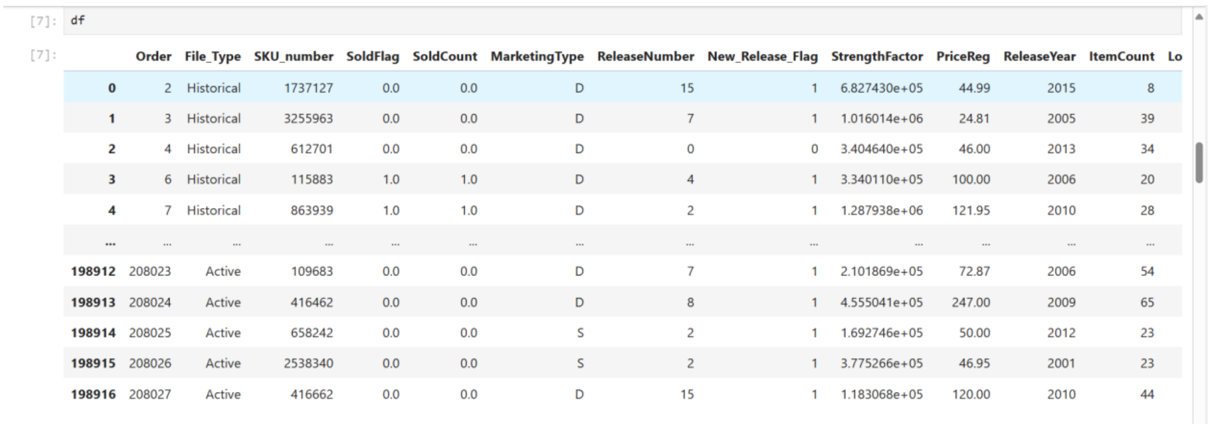
	Order	File_Type	SKU_number	SoldFlag	SoldCount	MarketingType	ReleaseNumber	New_Release_Flag	StrengthFactor	PriceReg	ReleaseYear	ItemCount	Lo
0	2	Historical	1737127	0.0	0.0	D	15	1	6.827430e+05	44.99	2015	8	
1	3	Historical	3255963	0.0	0.0	D	7	1	1.016014e+06	24.81	2005	39	
2	4	Historical	612701	0.0	0.0	D	0	0	3.404640e+05	46.00	2013	34	
3	6	Historical	115883	1.0	1.0	D	4	1	3.340110e+05	100.00	2006	20	
4	7	Historical	863939	1.0	1.0	D	2	1	1.287938e+06	121.95	2010	28	
...	...	...	...	...	...	...	...	...	...	...	...	...	
198912	208023	Active	109683	NaN	NaN	D	7	1	2.101869e+05	72.87	2006	54	
198913	208024	Active	416462	NaN	NaN	D	8	1	4.555041e+05	247.00	2009	65	
198914	208025	Active	658242	NaN	NaN	S	2	1	1.692746e+05	50.00	2012	23	



```
[4]: df.isnull().sum()
```

```
[4]: Order                    0
     File_Type                0
     SKU_number               0
     SoldFlag            122921
     SoldCount           122921
     MarketingType            0
     ReleaseNumber            0
     New_Release_Flag         0
     StrengthFactor           0
     PriceReg                 0
     ReleaseYear              0
     ItemCount                0
     LowUserPrice             0
     LowNetPrice              0
     dtype: int64
```

```
[5]: df['SoldFlag'].fillna(0, inplace=True)
     df['SoldCount'].fillna(0, inplace=True)
```



```
[7]: df
```

[7]:

	Order	File_Type	SKU_number	SoldFlag	SoldCount	MarketingType	ReleaseNumber	New_Release_Flag	StrengthFactor	PriceReg	ReleaseYear	ItemCount	Lo
0	2	Historical	1737127	0.0	0.0	D	15	1	6.827430e+05	44.99	2015	8	
1	3	Historical	3255963	0.0	0.0	D	7	1	1.016014e+06	24.81	2005	39	
2	4	Historical	612701	0.0	0.0	D	0	0	3.404640e+05	46.00	2013	34	
3	6	Historical	115883	1.0	1.0	D	4	1	3.340110e+05	100.00	2006	20	
4	7	Historical	863939	1.0	1.0	D	2	1	1.287938e+06	121.95	2010	28	
...	...	...	...	...	...	...	...	...	...	...	...	...	
198912	208023	Active	109683	0.0	0.0	D	7	1	2.101869e+05	72.87	2006	54	
198913	208024	Active	416462	0.0	0.0	D	8	1	4.555041e+05	247.00	2009	65	
198914	208025	Active	658242	0.0	0.0	S	2	1	1.692746e+05	50.00	2012	23	
198915	208026	Active	2538340	0.0	0.0	S	2	1	3.775266e+05	46.95	2001	23	
198916	208027	Active	416662	0.0	0.0	D	15	1	1.183068e+05	120.00	2010	44	

Exploratory Data Analysis (EDA):

Analyze sales trends and variations over time

2. Exploratory Data Analysis (EDA):

- **Analyze sales trends and variations over time**
- **Identify top-selling products and categories.**
- **Investigate stock levels and low-stock items.**

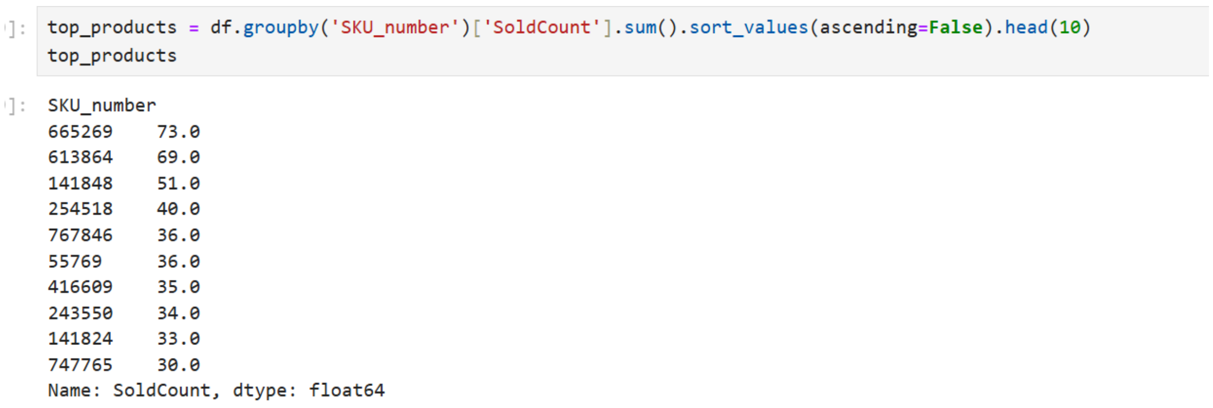
Analysing Sales trends

```
sales_by_year = df.groupby('ReleaseYear')['SoldCount'].sum()
```

```
sales_by_year
```

```
ReleaseYear
0         0.0
1900      0.0
1904      0.0
1905      0.0
1914      0.0
        ...
2014    794.0
2015    122.0
2016      3.0
2017      0.0
2018      0.0
Name: SoldCount, Length: 85, dtype: float64
```

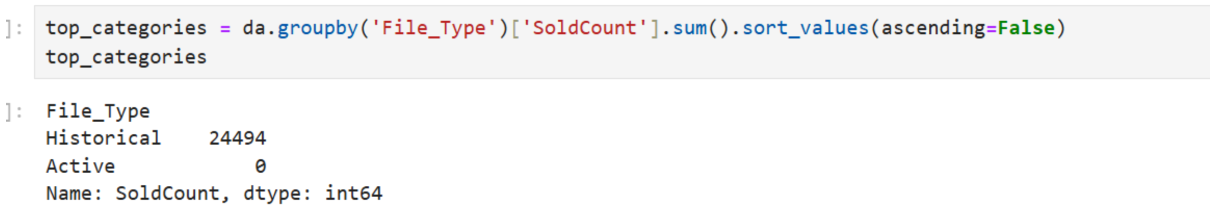
TOP 10 products



```
top_products = df.groupby('SKU_number')['SoldCount'].sum().sort_values(ascending=False).head(10)
top_products
```

```
SKU_number
665269    73.0
613864    69.0
141848    51.0
254518    40.0
767846    36.0
55769     36.0
416609    35.0
243550    34.0
141824    33.0
747765    30.0
Name: SoldCount, dtype: float64
```

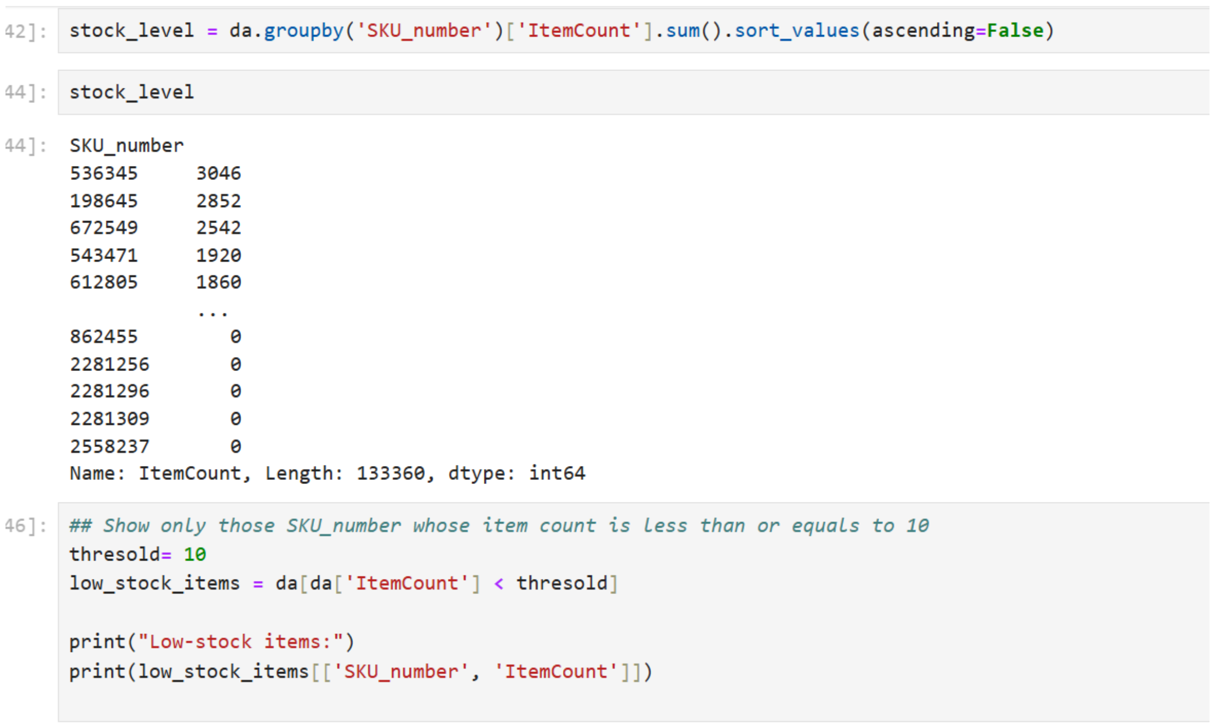
Top Categories



```
top_categories = da.groupby('File_Type')['SoldCount'].sum().sort_values(ascending=False)
top_categories
```

```
File_Type
Historical    24494
Active            0
Name: SoldCount, dtype: int64
```

Investigate stock levels and low-stock items.

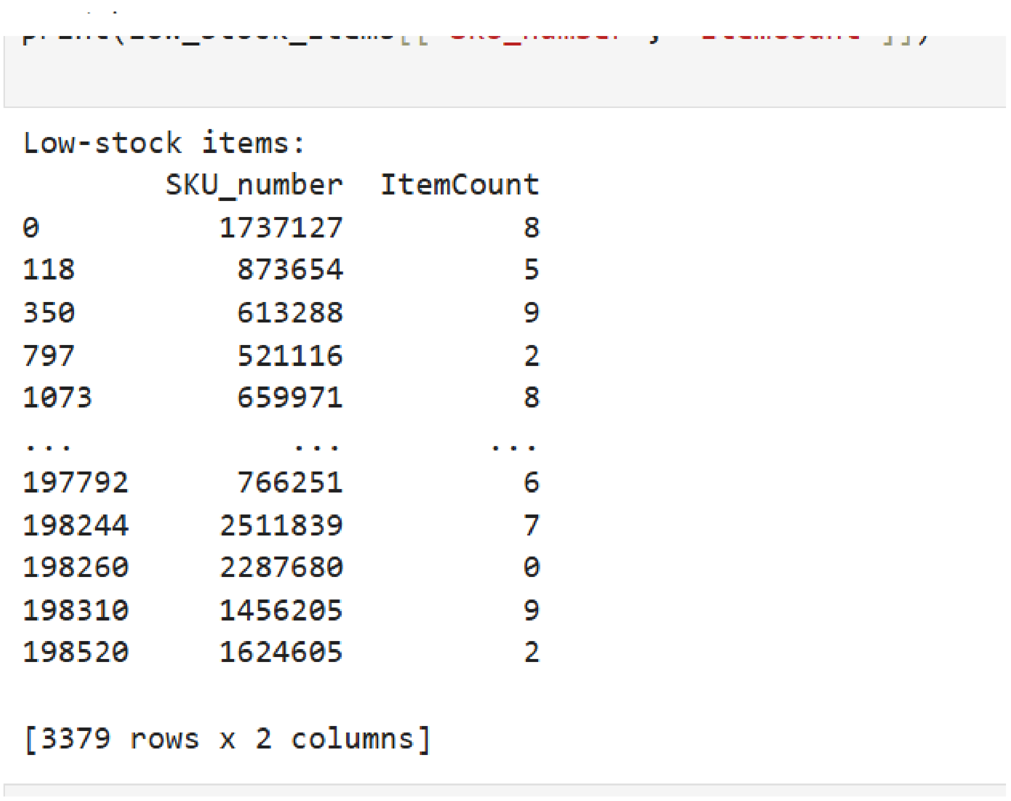


```
42]: stock_level = da.groupby('SKU_number')['ItemCount'].sum().sort_values(ascending=False)
```

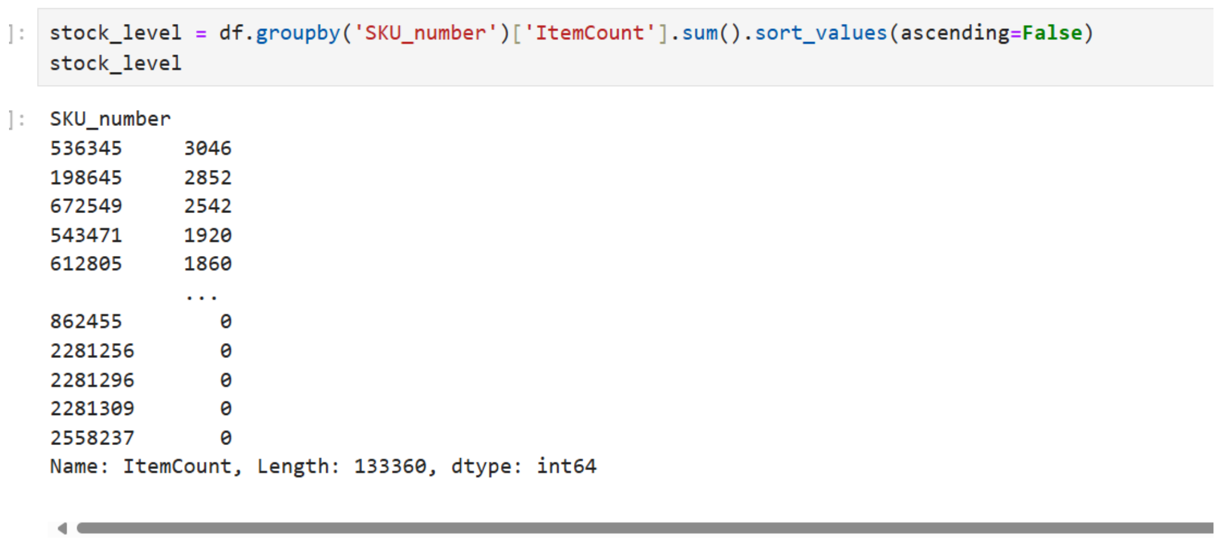
```
44]: stock_level
```

```
44]: SKU_number
     536345     3046
     198645     2852
     672549     2542
     543471     1920
     612805     1860
                ...
     862455        0
     2281256       0
     2281296       0
     2281309       0
     2558237       0
     Name: ItemCount, Length: 133360, dtype: int64
```

```
46]: ## Show only those SKU_number whose item count is less than or equals to 10
     thresold= 10
     low_stock_items = da[da['ItemCount'] < thresold]

     print("Low-stock items:")
     print(low_stock_items[['SKU_number', 'ItemCount']])
```

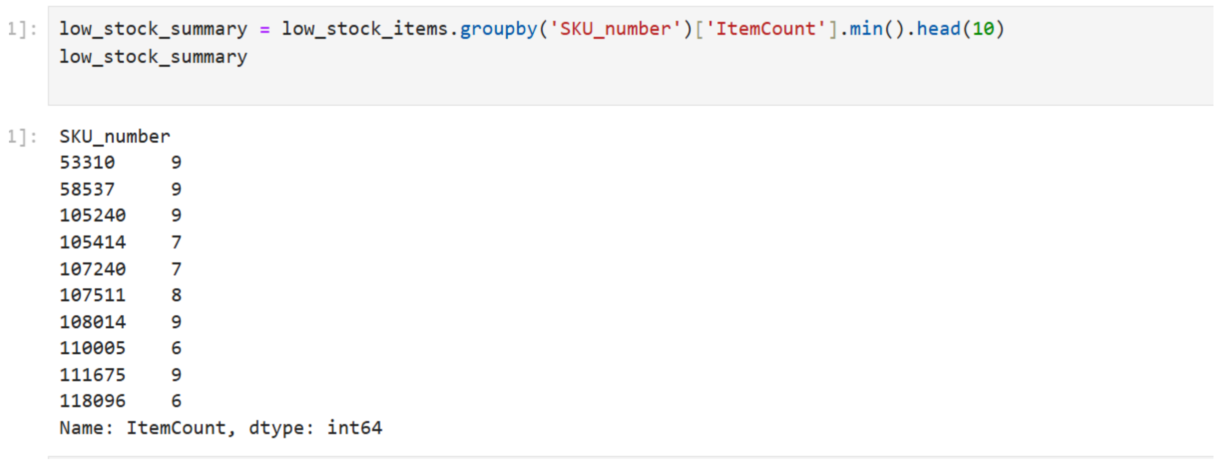


```
Low-stock items:
        SKU_number  ItemCount
0          1737127          8
118         873654          5
350         613288          9
797         521116          2
1073        659971          8
...            ...        ...
197792      766251          6
198244     2511839          7
198260     2287680          0
198310     1456205          9
198520     1624605          2

[3379 rows x 2 columns]
```



```
]: stock_level = df.groupby('SKU_number')['ItemCount'].sum().sort_values(ascending=False)
   stock_level
```

```
]: SKU_number
   536345     3046
   198645     2852
   672549     2542
   543471     1920
   612805     1860
              ...
   862455        0
   2281256       0
   2281296       0
   2281309       0
   2558237       0
   Name: ItemCount, Length: 133360, dtype: int64
```



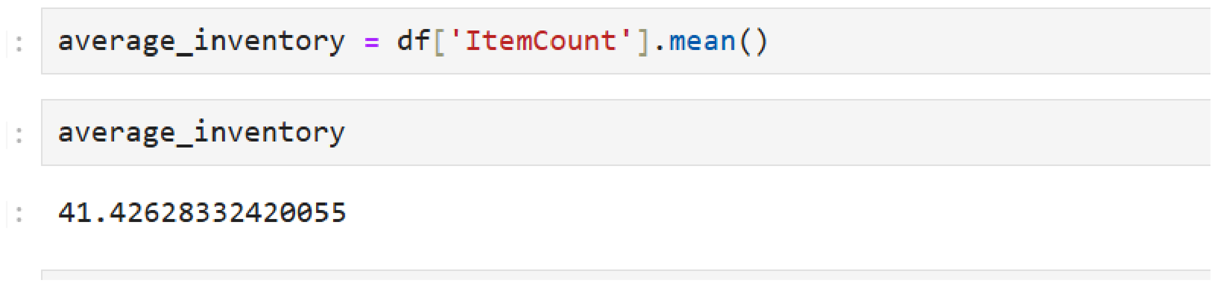
```
1]: low_stock_summary = low_stock_items.groupby('SKU_number')['ItemCount'].min().head(10)
    low_stock_summary
```

```
1]: SKU_number
    53310     9
    58537     9
    105240    9
    105414    7
    107240    7
    107511    8
    108014    9
    110005    6
    111675    9
    118096    6
    Name: ItemCount, dtype: int64
```

3.Inventory Insights and Recommendations:

1. Calculate key performance indicators (e.g., inventory turnover, stock-to-sales ratio, reorder points).

Average Inventory:

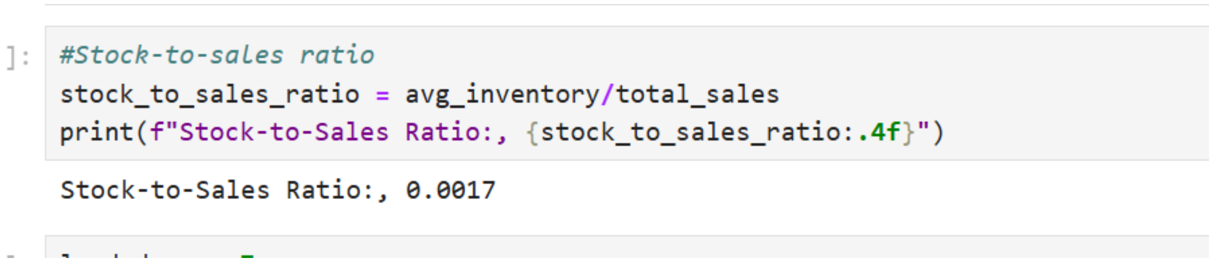


```
: average_inventory = df['ItemCount'].mean()
```

```
: average_inventory
```

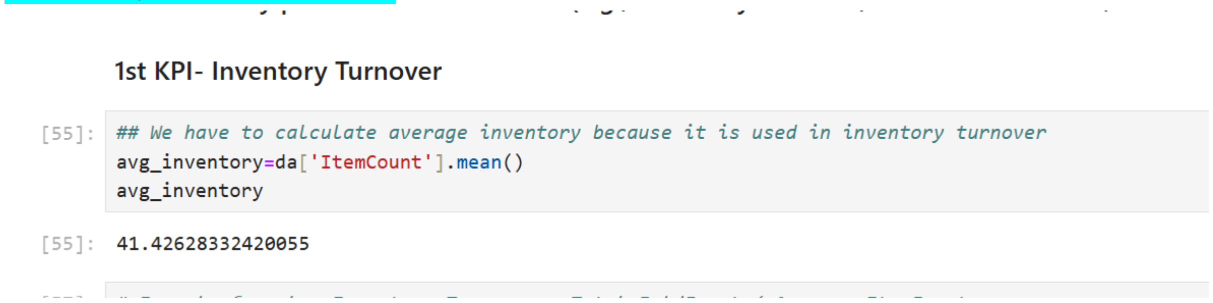
```
: 41.42628332420055
```

Stock To Sales Ratio:



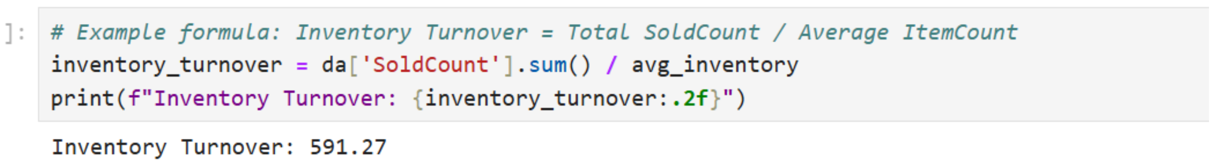
```
]: #Stock-to-sales ratio
   stock_to_sales_ratio = avg_inventory/total_sales
   print(f"Stock-to-Sales Ratio:, {stock_to_sales_ratio:.4f}")

   Stock-to-Sales Ratio:, 0.0017
```

Inventory Turnover Ratio:



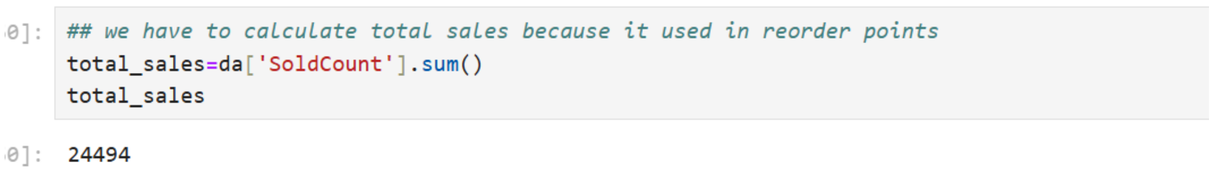
1st KPI- Inventory Turnover

```
[55]: ## We have to calculate average inventory because it is used in inventory turnover
      avg_inventory=da['ItemCount'].mean()
      avg_inventory

[55]: 41.42628332420055
```



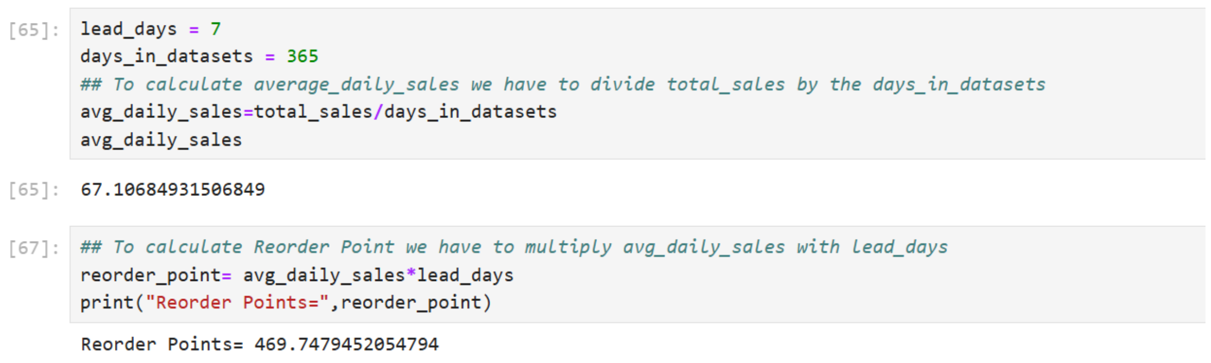
```
]: # Example formula: Inventory Turnover = Total SoldCount / Average ItemCount
   inventory_turnover = da['SoldCount'].sum() / avg_inventory
   print(f"Inventory Turnover: {inventory_turnover:.2f}")

   Inventory Turnover: 591.27
```

Total Sales:



```
0]: ## we have to calculate total sales because it used in reorder points
    total_sales=da['SoldCount'].sum()
    total_sales

0]: 24494
```

Reorder point:



```
[65]: lead_days = 7
      days_in_datasets = 365
      ## To calculate average_daily_sales we have to divide total_sales by the days_in_datasets
      avg_daily_sales=total_sales/days_in_datasets
      avg_daily_sales

[65]: 67.10684931506849

[67]: ## To calculate Reorder Point we have to multiply avg_daily_sales with lead_days
      reorder_point= avg_daily_sales*lead_days
      print("Reorder Points=",reorder_point)

      Reorder Points= 469.7479452054794
```

2. Provide actionable recommendations to optimize inventory management based on sales patterns.

1. Adjust Safety Stock Levels

- **Insight**: Identify products with fluctuating or high demand using historical sales trends.
- **Action:** Maintain higher safety stock levels for these items to prevent stockouts during demand spikes.
- **Benefit**: Improved customer satisfaction and reduced lost sales opportunities.

2. Automate Low-Stock Alerts

- **Insight:** Track low-stock items frequently and automate alerts when inventory reaches reorder points.
- **Action:** Set up an automated inventory management system to trigger restocking notifications.
- **Benefit:** Proactive replenishment prevents interruptions in the supply chain.

3.Focus on Top-Performing Products

- **Insight:** Top-selling products contribute significantly to revenue.
- **Action:** Allocate more inventory to these products and prioritize their availability.

- **Benefit:** Higher sales and improved profitability.

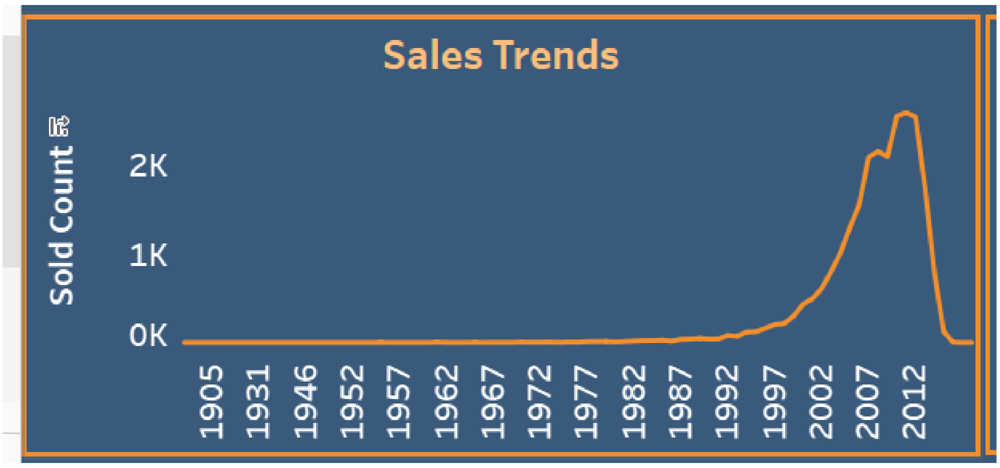
4. **Optimize Inventory Turnover**

- **Insight:** Calculate and monitor inventory turnover rates to assess efficiency.
- **Action:** Rotate inventory regularly, focusing on products with slower turnover rates.
- **Benefit:** Reduced holding costs and minimized risk of obsolescence.

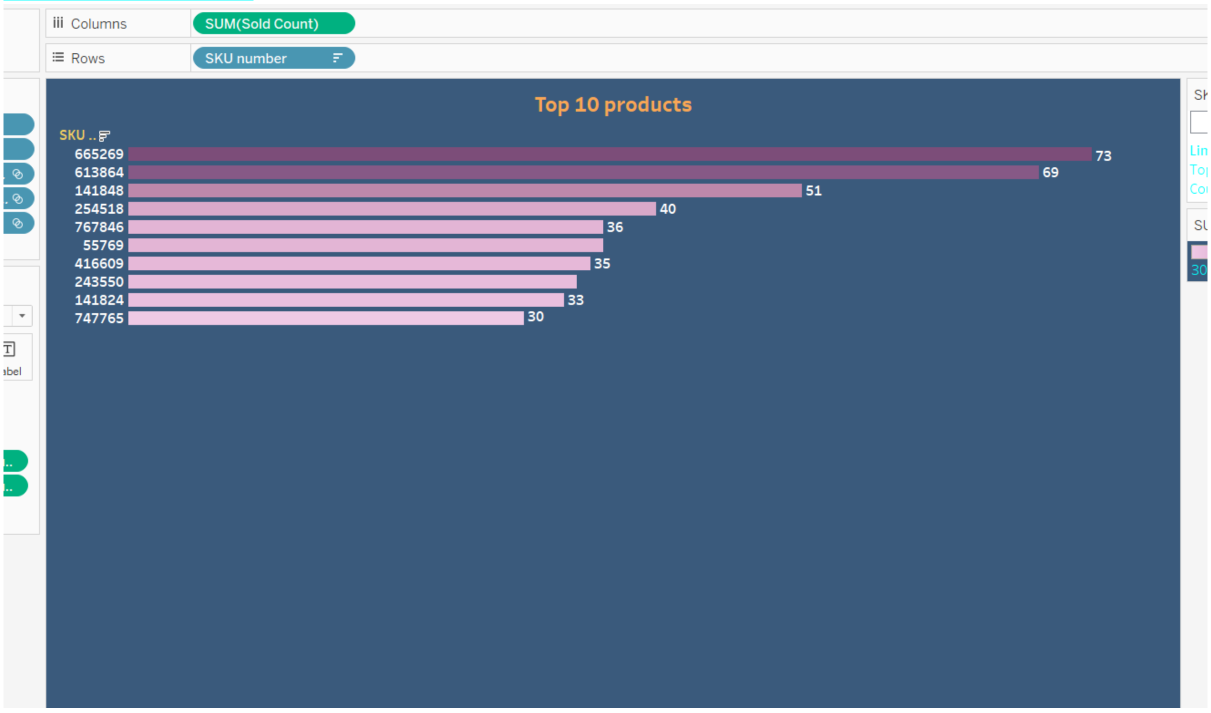
4. Data Visualization:

    1. Create interactive and informative visualizations (e.g., line charts, bar plots) to present sales trends and inventory metrics.
    2. Highlight insights through well-designed graphs and charts.

1.Sales Trends over time

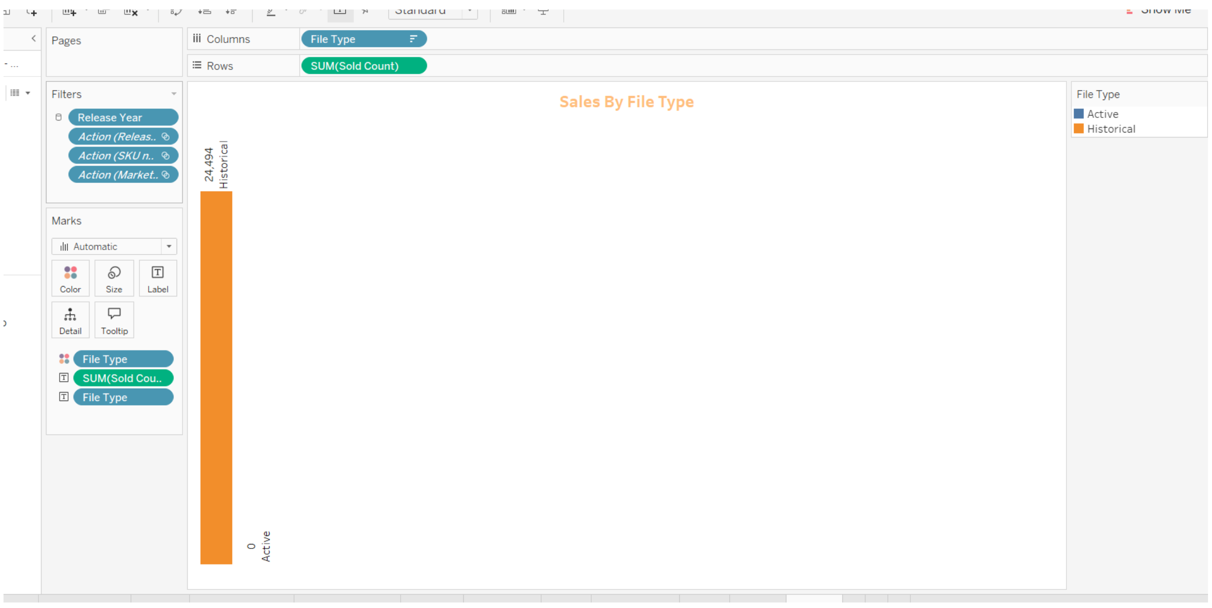


2. **Top 10 products**

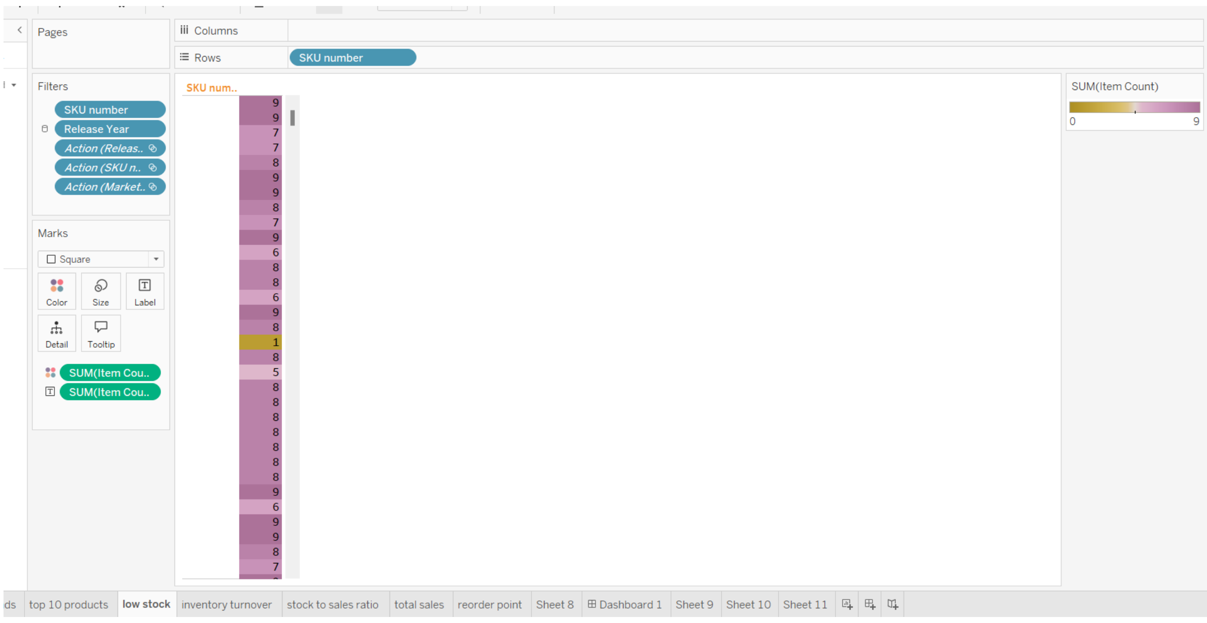


SKU	Sold Count
665269	73
613864	69
141848	51
254518	40
767846	36
55769	
416609	35
243550	
141824	33
747765	30

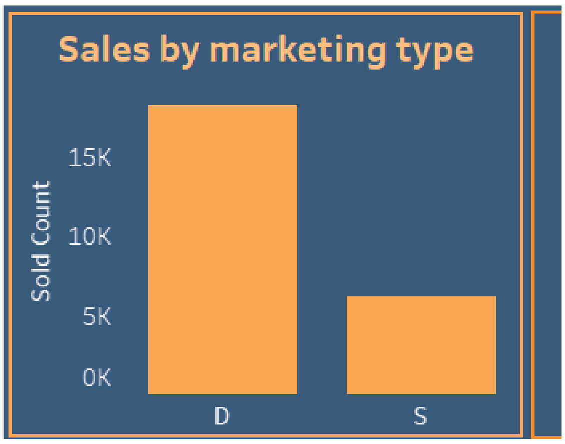
3.Top Category:



4 Low stock items

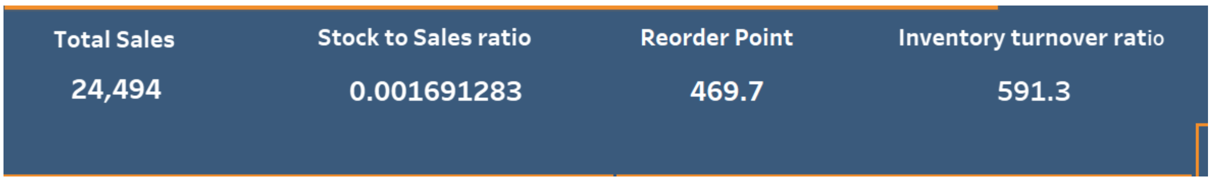


```
9
9
7
7
8
9
9
8
7
9
6
8
8
6
9
8
1
8
5
8
8
8
8
8
8
8
9
6
9
9
8
7
```



Inventory Metrics:

key performance indicators (e.g., inventory turnover, stock-to-sales ratio, reorder points and total sales).



Total Sales	Stock to Sales ratio	Reorder Point	Inventory turnover ratio
24,494	0.001691283	469.7	591.3

5 .Documentation and Reporting:

Summarize the findings, inventory-driven insights, and recommendations from the analysis.

Explain how the inventory-focused insights can benefit businesses in enhancing inventory management.

Summary of Findings:

1. **Sales Patterns and Trends:**
   - Seasonal Peaks: Sales data reveals significant spikes during specific months, indicating seasonal demand for certain products.
   - Consistent Top Performers: Certain products consistently rank as best-sellers, driving a large portion of revenue.
2. **Inventory Performance:**
   - Low-Stock Items: Identified products that frequently run out of stock, leading to potential lost sales.
   - Overstock Issues: Highlighted items with slow turnover rates, occupying storage space and increasing carrying costs.
3. **Key Performance Indicators (KPIs):**
   - **Inventory Turnover Ratio**: Indicates the speed at which inventory is sold and replaced. Some categories exhibit inefficient turnover rates.

     Stock-to-Sales Ratio: Identifies mismatches between stock levels and actual demand.

     Reorder Point: The reorder point 469.7 tells us that When the inventory level of that product drops to approximately 470 units, it is time to place a new order.

- **Explain how the inventory-focused insights can benefit businesses in enhancing inventory management.**

1 Optimized Stock Levels:

Explanation: By analyzing sales patterns and demand forecasts, businesses can maintain optimal stock levels. This ensures that they have enough inventory to meet customer demand without holding excess stock, which ties up capital and incurs storage costs.

2 Cost Reduction:

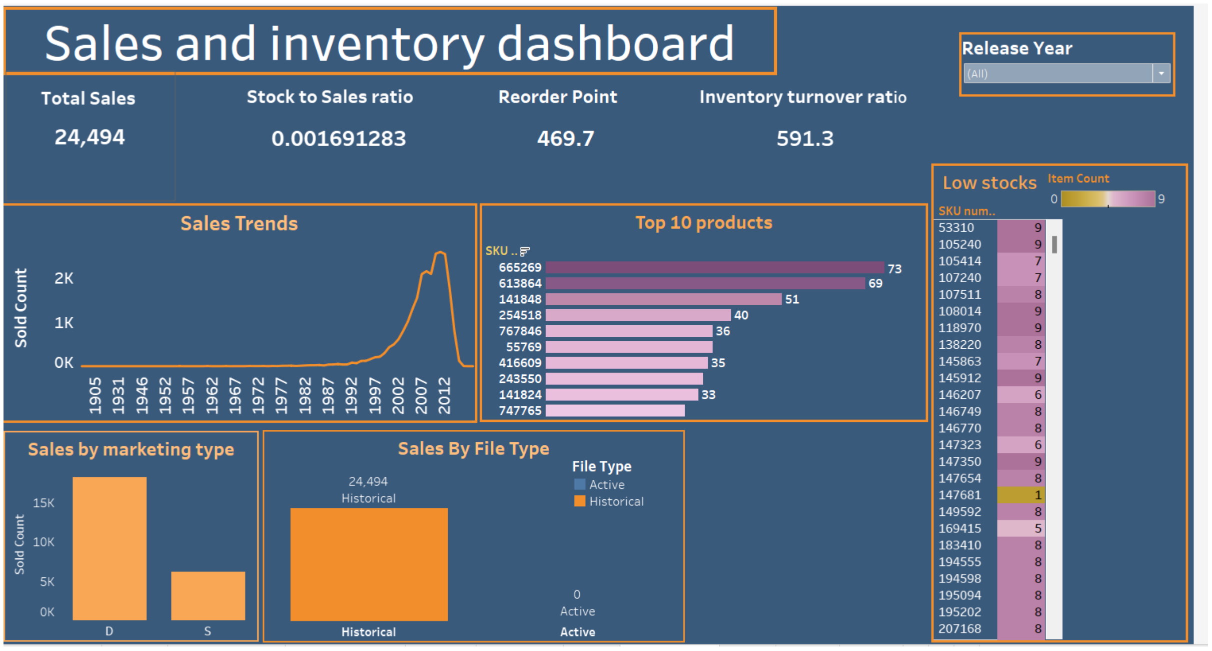
Explanation: Efficient inventory management minimizes carrying costs, reduces wastage, and optimizes the use of warehouse space. This leads to significant cost savings, enhancing overall profitability.

3. Accurate Reorder Points:

Prevention of stockouts and overstocking.

Explanation: Insights from sales data and inventory turnover rates help in setting accurate reorder points. This ensures timely replenishment of stock, preventing both stockouts and excess inventory.

Utilize advanced visualization tools like Plotly or Tableau for interactive visualizations.


Sales and inventory dashboard

Release Year: (All)

Total Sales	Stock to Sales ratio	Reorder Point	Inventory turnover ratio
24,494	0.001691283	469.7	591.3

Low stocks

SKU num..	Item Count
53310	9
105240	9
105414	7
107240	7
107511	8
108014	9
118970	9
138220	8
145863	7
145912	9
146207	6
146749	8
146770	8
147323	6
147350	9
147654	8
147681	1
149592	8
169415	5
183410	8
194555	8
194598	8
195094	8
195202	8
207168	8

Top 10 products

SKU ..	
665269	73
613864	69
141848	51
254518	40
767846	36
55769	
416609	35
243550	
141824	33
747765	